Nushell Structured data shell

CONCEPTS

Pipelines pass typed data (tables, records, lists), not text, so filter by column instead of parsing with grep and awk.

Source ls, open, ps
↓ |
Table rows + columns
↓ `where / select`
Filtered still structured
↓ `to json / save`
Output text or file

Nushell is not POSIX: no $VAR, no &&, no backtick substitution. External commands still run and return text.

NAVIGATE & FILES

`ls`
List files as a table

`ls **/*.rs`
Glob recursively

`cd <dir>`
Change directory

`mkdir <dir>`
Create directory with parents

OPEN & CONVERT

`open <file>`
Load file, parsed by extension

`open --raw <file>`
Read as plain text

`save <file>`
Write pipeline to file

`from json / from yaml / from csv`
Parse text into data

`to json / to yaml / to csv`
Render data as text

`http get <url>`
Fetch URL, parse JSON

FILTER & SHAPE

`where <col> == <val>`
Keep matching rows

`select <col> <col>`
Keep columns

FILTER & SHAPE (cont.)

`reject <col>`
Drop columns

`get <col>`
Pull out column or field

`get a.b.0`
Cell path into nested data

`sort-by <col>`
Sort rows

`sort-by -r <col>`
Sort descending

`first <n>` / `last <n>`
Take rows from either end

`each {|x| ... }`
Map over items

`update <col> {|r| ... }`
Change a column

`insert <col> {|r| ... }`
Add a new column

STRINGS & TEXT

`$"hello ($name)"`
Interpolate expressions

`str replace <a> <b>`
Replace text

`split row <sep>`
String to list

`lines`
Split text into lines

VARIABLES & FLOW

`let <name> = <value>`
Immutable variable

`mut <name> = <value>`
Mutable variable

`if <cond> { } else { }`
Conditional (an expression)

`for i in 1..5 { print $i }`
Loop over range or list

COMMANDS & MODULES

`def <name> [x: int] { }`
Custom command

`help <command>`
Help for one command